**Practical no -8**

**Pincode.java**

import java.io.Serializable;

import javax.persistence.Basic;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.NamedQueries;

import javax.persistence.NamedQuery;

import javax.persistence.Table;

import javax.validation.constraints.Size;

import javax.xml.bind.annotation.XmlRootElement;

@Entity

@Table(name = "pincode")

@XmlRootElement

@NamedQueries({

@NamedQuery(name = "Pincode.findAll", query = "SELECT p FROM Pincode p")

, @NamedQuery(name = "Pincode.findByPid", query = "SELECT p FROM Pincode p WHERE p.pid = :pid")

, @NamedQuery(name = "Pincode.findByPcode", query = "SELECT p FROM Pincode p WHERE p.pcode = :pcode")

, @NamedQuery(name = "Pincode.findByState", query = "SELECT p FROM Pincode p WHERE p.state = :state")

, @NamedQuery(name = "Pincode.findByCity", query = "SELECT p FROM Pincode p WHERE p.city = :city")})

public class Pincode implements Serializable {

private static final long serialVersionUID = 1L;

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Basic(optional = false)

@Column(name = "pid")

private Integer pid;

@Column(name = "pcode")

private Integer pcode;

@Size(max = 20)

@Column(name = "state")

private String state;

@Size(max = 20)

@Column(name = "city")

private String city;

public Pincode() {

}

public Pincode(Integer pid) {

this.pid = pid;

}

**Roll no -17,19**

public Integer getPid() {

return pid;

}

public void setPid(Integer pid) {

this.pid = pid;

}

public Integer getPcode() {

return pcode;

}

public void setPcode(Integer pcode) {

this.pcode = pcode;

}

public String getState() {

return state;

}

public void setState(String state) {

this.state = state;

}

public String getCity() {

return city;

}

public void setCity(String city) {

this.city = city;

}

@Override

public int hashCode() {

int hash = 0;

hash += (pid != null ? pid.hashCode() : 0);

return hash;

}

@Override

public boolean equals(Object object) {

// TODO: Warning - this method won't work in the case the id fields are not set

if (!(object instanceof Pincode)) {

return false;

}

Pincode other = (Pincode) object;

if ((this.pid == null && other.pid != null) || (this.pid != null && !this.pid.equals(other.pid))) {

return false;

}

return true;

}

@Override

public String toString() {

return "entityResource.Pincode[ pid=" + pid + " ]";}}

**PincodeResource.java**

import javax.persistence.EntityManager;

import javax.persistence.EntityManagerFactory;

import javax.persistence.Persistence;

import javax.persistence.PersistenceContext;

import javax.ws.rs.core.Context;

import javax.ws.rs.core.UriInfo;

import javax.ws.rs.Produces;

import javax.ws.rs.Consumes;

import javax.ws.rs.GET;

import javax.ws.rs.Path;

import javax.ws.rs.PUT;

import javax.ws.rs.PathParam;

import javax.ws.rs.core.MediaType;

import org.json.JSONArray;

import org.json.JSONException;

import org.json.JSONObject;

/\*\*

\* REST Web Service

\*

\* @author Pramod SJ

\*/

@Path("pincode")

public class PincodeResource {

@Context

private UriInfo context;

@PersistenceContext(unitName="PincodeApiPU")

private final EntityManager em;

private final EntityManagerFactory emf;

Connection conn;

Statement stmt;

/\*\*

\* Creates a new instance of PincodeResource

\*/

public PincodeResource() {

emf=Persistence.createEntityManagerFactory("PincodeApiPU");

em=emf.createEntityManager();

}

/\*\*

\* Retrieves representation of an instance of service.PincodeResource

\* @return an instance of java.lang.String

\*/

@GET

@Produces(MediaType.APPLICATION\_JSON)

public String getPincode() throws JSONException {

List<Pincode> pincodes=em.createNamedQuery("Pincode.findAll").getResultList();

JSONObject mainObj=new JSONObject();

JSONArray array=new JSONArray();

for(Pincode pincode:pincodes){

JSONObject obj=new JSONObject();

obj.put("pid", pincode.getPid());

obj.put("pcode",pincode.getPcode());

obj.put("state", pincode.getState());

obj.put("city",pincode.getCity());

array.put(obj);

}

mainObj.put("pincodes",array);

return mainObj.toString();

}

@GET

@Path("/pincode/findDataByPincode/{pincode}")

@Produces(MediaType.APPLICATION\_JSON)

public String getData(@PathParam("pcode") String id) throws JSONException, SQLException {

List<Pincode> pincodes=em.createNamedQuery("Pincode.findByPcode").setParameter("pcode", id).getResultList();

JSONObject mainObj=new JSONObject();

JSONArray array=new JSONArray();

for(Pincode pincode:pincodes){

JSONObject obj=new JSONObject();

obj.put("pcode",String.valueOf(pincode.getPcode()));

obj.put("state", pincode.getState());

obj.put("city",pincode.getCity());

array.put(obj);

}

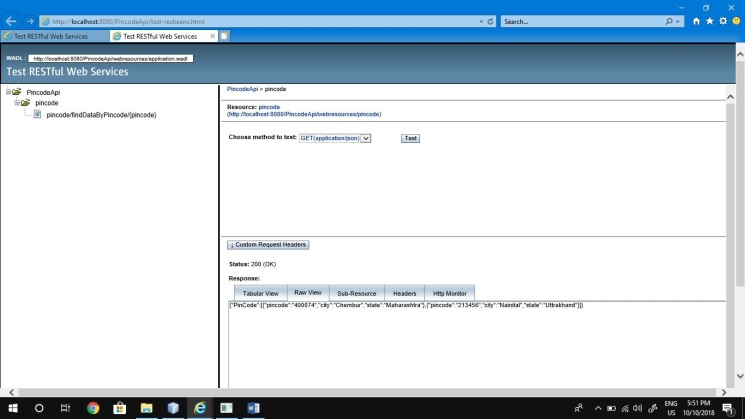
mainObj.put("pincodes",array);

return mainObj.toString();

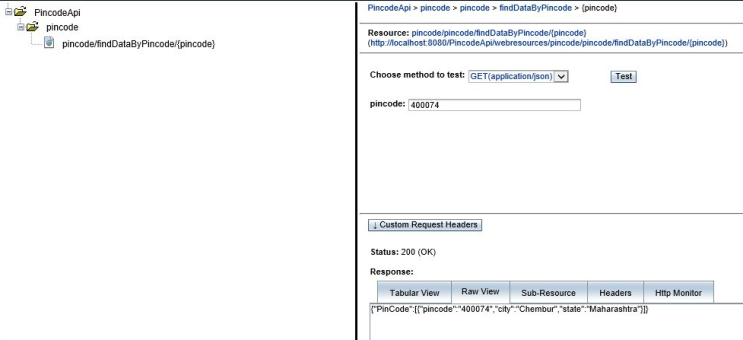
}

}

**Output:**

****

**Input as Pincode:**

****